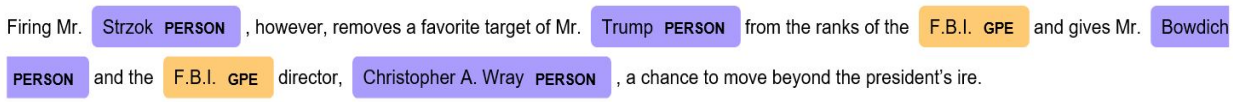
Medical Entity Recognition

Team No 5
Aditya Saripalli (20173071)
Tarun Vatwani (2018201075)
Geethika Dudyala (20161154)
Aman Bansal (20161008)

Abstract & Problem Statement

Medical Entity Recognition(MER) is an application of a very famous problem in the field of Information Extraction(IE) i.e. Named Entity Recognition(NER). In other words it is a domain specific NER in this case the domain is medical field.

NER is a method/model which takes a series of text(sentences or paragraphs) and provides labels to noun phrases present in that piece of text. The image below shows an example of how NER model typically works.



This project aims at parsing named entities and in this project, we have to recognize and classify medical data into the relevant categories, namely drugs, diseases, symptoms, side-effects, treatment, etc. Twitter data will be the input and based on previous medical data from databases and ontologies, relevant medical terms have to be parsed and classified (medical named entities are recognized and classified based on the category they belong to (ex: drug or a disease or cure etc....)

As the name suggests, a Medical Name Entity Recognizer identifies medical entities in text. Medical entities, in the context of our project, are fixed, there are 5 categories as mentioned above. Previously, researchers in the field have used hand crafted features to identify medical

entities in medical literature. In this project, we have to extend medical entity recognition on tweets. We would use NLP toolkits designed for processing tweets along with other medical ontologies (or databases) to exploit semantic features for this task. We attempt to push the accuracy beyond what has been obtained so far. As tweets are highly disorganised and prone to inconsistencies, our work includes filtering out all these inconsistencies.

Related Work

With the progress in natural language processing (NLP), extracting valuable information from biomedical literature has gained popularity among researchers, and deep learning has boosted the development of effective biomedical text mining models. However, directly applying the advancements in NLP to biomedical text mining often yields unsatisfactory results due to a word distribution shift from general domain corpora to biomedical corpora.

A lot of research groups have proposed various approaches for medical entity recognition based on deep learning in recent years but all those approaches are trained on Electronic Health Records(EHRs). The most recent work done in Medical Entity Recognition is BioBERT which is the first domain-specific BERT based model pre-trained on biomedical corpora (PubMed abstracts and PMC full-text articles) which happens to be state of the art approach for MER. BioBERT has a simple architecture based on bidirectional trans-formers. BERT uses a single output layer based on the representa-tions from its last layer to compute only token level BIO2 probabilities. Note that while previous works in biomedical NERoften used word embeddings trained on PubMed or PMC corpora(Habibiet al., 2017;Yoonet al., 2019), BioBERT directly learnsWordPiece embeddings during pre-training and fine-tuning [1].

Previously proposed neural network architectures included hybrid convolutional neural network (CNN) and bi-directional long-short term memory (Bi-LSTM) as introduced by [2]. Previous State-of-the-art NER models used the architecture proposed in [3], a stacked bi-directional long-short term memory(Bi-LSTM) for both character and word, with a CRF layer on top of the network.

Long Short-Term Memory (LSTM), have been widely used for deriving contextual features for CRF model training and have demonstrated promising per-formance for medical entity recognition. Although RNN models provide a good set of features for NER, they depend heavily on the word embedding models which are not good at dealing with the complex characteristics of word use under different linguistic contexts. To incorporate that several approaches have

been proposed using the word representations of the ELMo which contain richer information compared to a standard word embedding such as Word2Vec [4].

Datasets

1. **CADEC dataset :** It is a corpus of medical forum posts on patient-reported Adverse Drug Events (ADEs).
    - ★ The corpus is sourced from posts on social media, and contains text that is largely written in colloquial language and often deviates from formal English grammar and punctuation rules.
    - ★ Annotations contain mentions of concepts such as drugs, adverse effects, symptoms, and diseases linked to their corresponding concepts in controlled vocabularies.
2. **Twimed Dataset**
3. **Micromed Dataset :** This dataset was described in MedInfo 2015 paper from IBM Melbourne Research lab.
    - ★ consists of tweet annotations with medical entities. (three types of entities: Disease (T047 in UMLS), Symptoms (T184), and Pharmacologic Substance (T121)

The above two datasets are already available but these datasets were very small in size, hence not sufficient for training.
We have increased the size of the dataset that is available for training heuristically.

For this, we were given three resources,

R1 : A list of hashtags, which are relevant to the medical domain
R2 : A general tweet corpus (about 40-50 GB in size)
R3 : A list of medical terms and their appropriate categories.

Using the list of hashtags (R1) we obtained a new dataset consisting of a subset of the general tweet corpus (R2) - medical domain related tweets. This new dataset was used both for training as well as testing purpose.
A part of this new dataset can be annotated using R3 and was used for training purposes. The rest of the dataset was used for testing purposes.

Methodologies Tried

1. **Dataset Preprocessing:** For this phase we have worked with three datasets. We were provided with two datasets i.e. CADEC and Micromed.

   We found another dataset TwiMed for our training, it is a corpus of text from PubMed and Twitter which is annotated by two pharmacists, comprises semantically correct annotations for a set of drugs, diseases, and symptoms. This corpus contains the annotations for 3144 entities, 2749 relations, and 5003 attributes.

   The datasets available to us did not contain all the desired labels. CADEC is annotated with 5 labels (Drug, Disease, Symptom, Finding and Adverse Drug Reactions) whereas TwiMed and Micromed are labeled with Drug/Pharmacologic Substance, Disease and Symptoms. So we have trained with the given labels. We hope to train on desired labels in the next phase with generalized twitter corpus.

   We used the following approaches to preprocess our datasets:

   - ★ CADEC and TwiMed are annotated using brat annotation tool so to convert it into a dataframe we use the following script. The script gives output in ConLL data format from which it is converted into CSV file using a custom script. For twitter part of TwiMed it is again brat annotated .ann file, with named on their tweet IDs, so using their names we fetched out tweets(only 521 were present), and created a .txt file for each tweet

   - ★ MicroMed dataset is of .linejson format, with tweet IDs as keys, and ‘type’ contains type of annotation that word belongs to, and an attribute of json type named ‘location’ has two keys ‘start’ and ‘end’, which tells us that from where in text that word is starting and where it is ending. But in this field, data has very discrepancy, as the pointers are very different from the actual occurrences of words, for this we have to go manually file by file, where it occurs, to resolve it, and after doing all this we got only 370 tweets, as many of them were either deleted or account from which they were done is deactivated. After doing all this again we had converted it to ‘brat-flavored standoff’ format, as we had a ready script to get CSV from ‘brat-flavored standoff’ format, and after doing all this preprocessing we got ready to use CSV format of Twitter data.

2. **Twitter Data Processing:** This was one of the major challenges in phase 2. Handling such a large corpus and extracting and annotating was . Twitter archive file (.json format) have a specific dictionary structure, which contains various key value pairs describing the tweet. Out of all the key value pairs we needed the twitter text/extended text fileds, which are in english language, however extracting the text out of twitter json feeds also gives huge no of emoticons and other junk data which is not ASCII. So cleaning up the text was another major challenge.
   We wrote two scripts:
      1. SymptomsCorpusGenerator.py
      2. MedicalTweetsExtracter.py

   Script-1 generates the symptom corpus using symptom ids and Snomed symptoms descriptions.
   Script-2 clean ups and extracts the medical tweets(related to symptoms built above) and creates annotated data files.
   As the twitter corpus has files (closed to 2GB) saved everyday. So which amounts close to 60 GB(compressed). To extract all the relevant tweets from this huge corpus we took the approach of multiprocessing. We spawned 10 processes which would have a shared queue for Inter Process Communication. The shared queue has list of all the files in the uncompressed corpus.

3. **Model Preparation:** MER can be implemented as a sequence classification task, where every chunk is predicted IOB-style as Drug, Disease, Symptom, Treatment and Test. The IOB format (short for inside, outside, beginning) is a common tagging format for tagging tokens. The B- prefix before a tag indicates that the tag is the beginning of a chunk, and an I- prefix before a tag indicates that the tag is inside a chunk. The B- tag is used only when a tag is followed by a tag of the same type without O tokens between them. An O tag indicates that a token belongs to no chunk.

   We approached the problem of identifying the medical entities in the dataset using the following approaches – A naïve max-word-frequency approach, Conditional Random Fields (CRF), Long Short Term Memory (LSTM) and combination of LSTM-CRF. The LSTM-CRF is used by most of the state-of-the-art approaches to named entity recognition.

   The LSTM-CRF based approach is mentioned below:

We are given an input sequence $x = (x_1, \ldots, x_m)$, i.e. the words of a sentence and a sequence of output states $s = (s_1, \ldots, s_m)$, i.e. the named entity tags. In conditional random fields we model the conditional probability of the output state sequence given an input sequence given by:

$$p(s_1, s_2, ..., s_m | x_1, x_2, ..., x_m)$$

We did this by defining a feature map that maps an entire input sequence x paired with an entire state sequence s to some d-dimensional feature vector. Then we can model the probability as a log-linear model with the parameter vector $w \in \mathbb{R}^d$

$$p(s|x; w) = \frac{\exp(w \cdot \Phi(x, s))}{\sum_{s'} \exp(w \cdot \Phi(x, s'))},$$

$$w \cdot \Phi(x, s) = \text{score}_{crf}(x, s)$$

where s' ranges over all possible output sequences. We can view the expression as a scoring how well the state sequence fits the given input sequence. The idea is now, to replace the linear scoring function by a non-linear neural network. So we define the

$$\text{score}_{lstm-crf}(x, s) = \sum_{i=0}^{n} W_{s_{i-1}, s_i} \cdot \text{LSTM}(x)_i + b_{s_{i-1}, s_i},$$

score where W_{s_{i-1}, s_i} and b are the weight vector and the bias corresponding to the transition from s_{i-1} to s_i, respectively.

After constructing this score function, we can optimize the conditional probability and propagating back through the network. We are going to use the implementation provided by the keras-contrib package that contains useful extensions to the official keras package.

Architecture

- CRF

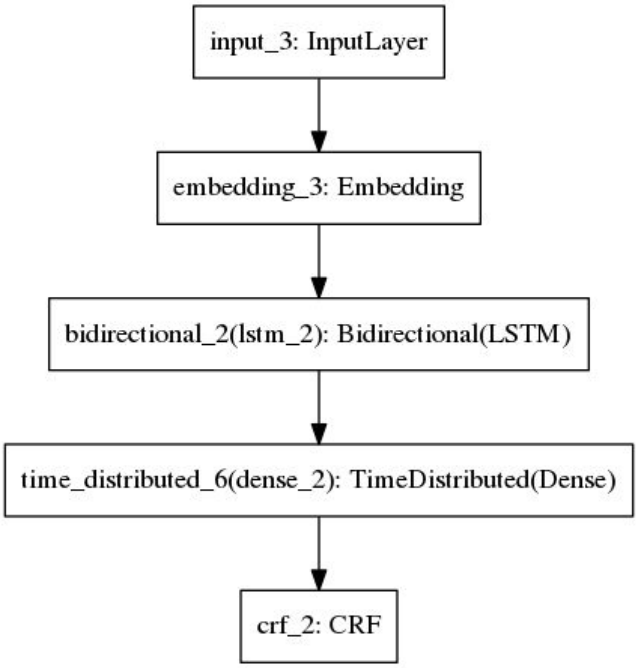
  Conditional random fields (CRFs) are a class of statistical modeling method is often applied in pattern recognition and machine learning and used for structured prediction. CRFs fall into the sequence modeling family. Whereas a discrete classifier predicts a label for a single sample without considering "neighboring" samples, a CRF can take context into account; e.g., the linear chain CRF (which is popular in natural language processing) predicts sequences of labels for sequences of input samples. We have explained the basic approach.

- LSTM

  We have used Keras to implement a bidirectional multi layer rnn cell (LSTM). The hyper parameters are present at the top in the jupyter notebook. Tweaking the parameters can yield a variety of results which are worth noting. We have used a softmax layer as the last layer of the network to produce the final classification outputs. We tried working with different optimizers and we found that AdamOptimzer produced the best results. To calculate the F1 Scores, Prediction Accuracy and Recall we use seqeval library.

- LSTM-CRF

  To improve upon the previous models we tried combining both LSTM and CRF so as to get the benefits of both. LSTM is used to replace the linear scoring function of CRF so as to get a non-linear and more expressive function. CRF on the other hand helps in detecting complicated named entities which is predominant in medial entities.



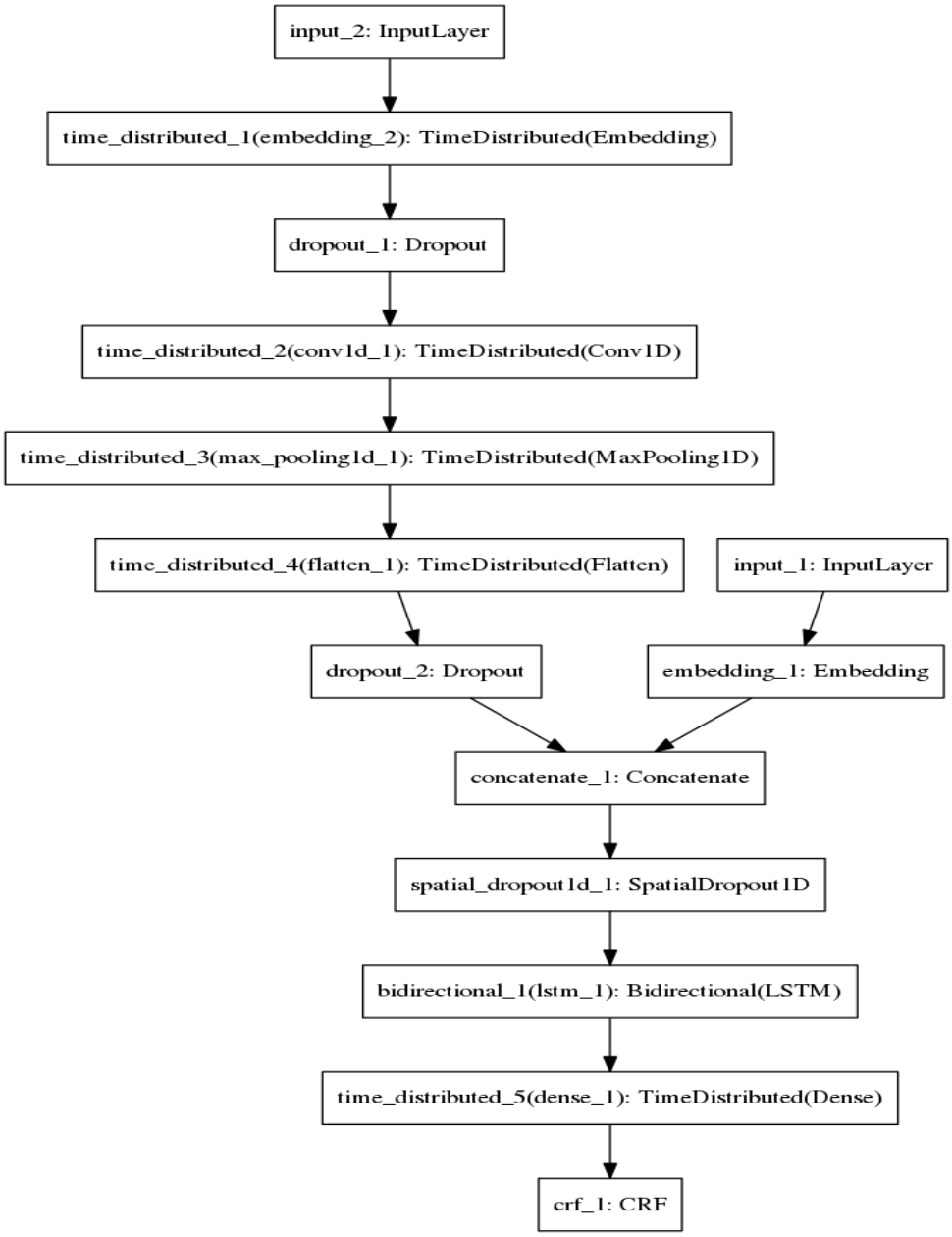
- LSTM-CRF with Character Embeddings

  Above we used LSTM on word level only but often much can be inferred from prefix and postfix of a word as well and so in this model. This is particularly useful when we have rare words which is very common in medical domain such as drugs, disease. This also helps in handling unknown words which may pop up at inference time.

  To encode the character-level information, we will use character embeddings and a LSTM to encode every word to a vector. We can use basically everything that produces a single vector for a sequence of characters that represent a word. Then we feed the vector to the main Bi-LSTM together with the learned word embeddings.

- CNN-LSTM-CRF with Character Embeddings

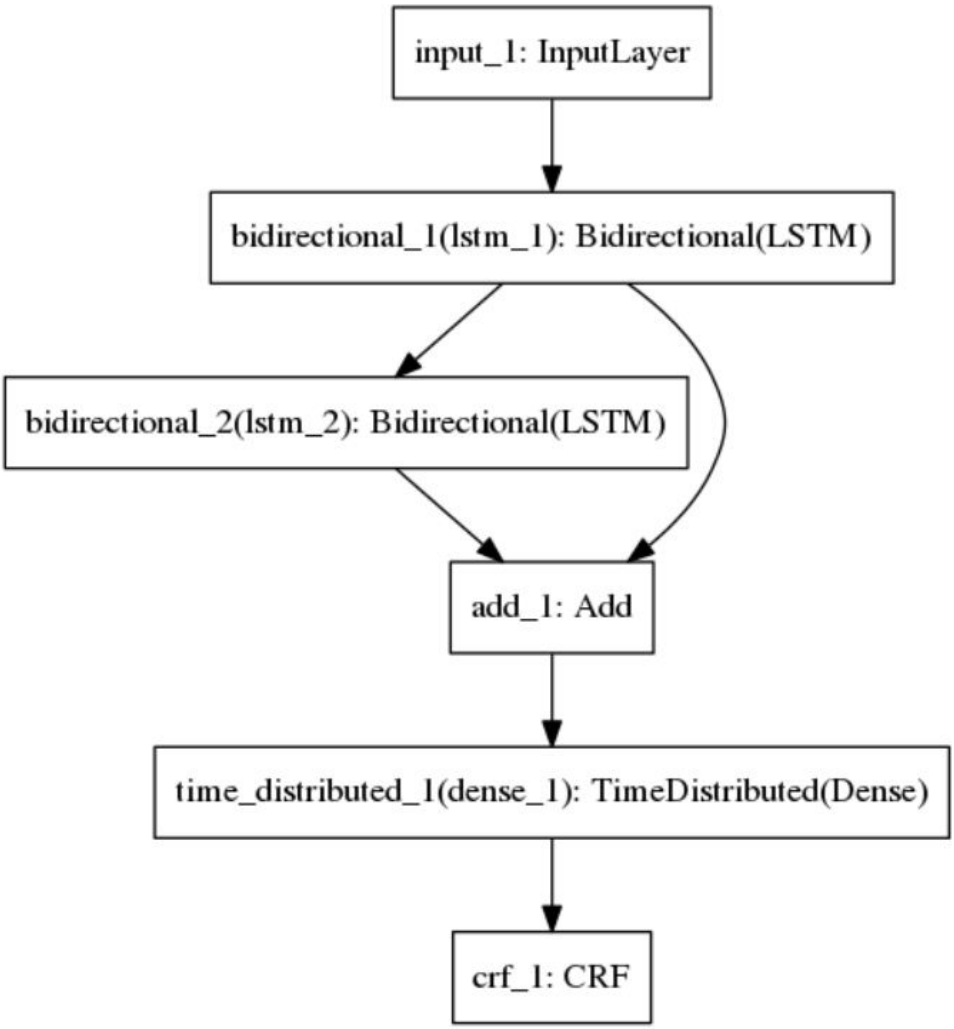
  Continuing with the model above we have added a CNN architecture along with the character embeddings and before we combine both the word embeddings and character encodings which is then fed to the main Bi-LSTM. Since medical data is sparse, we have to have a model which can predict the labels even with the small amount of data fed to it. This also adds additional expressivity to the DNN model and provides us with a much more powerful model for doing Medical Entity Recognition.



- Residual LSTM-CRF with ELMo

  Unlike most widely used word embeddings, ELMo word representations are functions of the entire input sentence. They are computed on top of two-layer bidirectional language model with character convolutions, as a linear function of the internal network states.

Concretely, ELMos use a pre-trained, multi-layer, bi-directional, LSTM-based language model and extract the hidden state of each layer for the input sequence of words. Then, they compute a weighted sum of those hidden states to obtain an embedding for each word. The weight of each hidden state is task-dependent and is learned. ELMo improves the performance of models across a wide range of tasks, spanning from question answering and sentiment analysis to named entity recognition. This setup allows us to do semi-supervised learning, where the biLM is pre-trained at a large scale and easily incorporated into a wide range of existing neural NLP architectures.



Evaluation Mechanism and Results

The Language-Independent Named Entity Recognition task introduced at CoNLL-2003 measures the performance of the systems in terms of precision, recall and f1-score, where:

Precision is the percentage of named entities found by the learning system that are correct. Recall is the percentage of named entities present in the corpus that are found by the system. A named entity is correct only if it is an exact match of the corresponding entity in the data file."

The following results are on CADEC, TwiMed and Micromed Dataset(combined). One of the issues with all these datasets is that these are annotated by different people so a concept in one dataset could be annotated differently in another. But we have assumed that are concepts are annotated uniformly across datasets.

LSTM-CRF with ELMo

F1-score: 68.4%

Entity	Precision	Recall	f1-score	Support
ADR	0.65	0.71	0.68	1122
Drug	0.95	0.90	0.92	335
Finding	0.30	0.37	0.33	90
Disease	0.67	0.10	0.17	62
Symptom	0.53	0.14	0.22	58

Micro Avg	0.68	0.68	0.68	1667
Macro Avg	0.69	0.68	0.67	1667

CNN-LSTM-CRF with Character Embeddings

F1-score: 61.6%

Entity	Precision	Recall	f1-score	Support
ADR	0.61	0.60	0.60	618
Drug	0.91	0.86	0.89	169
Finding	0.05	0.03	0.04	35
Disease	0.25	0.14	0.18	22
Symptom	0.17	0.05	0.08	37
Micro Avg	0.64	0.59	0.62	881
Macro Avg	0.62	0.59	0.60	881

LSTM-CRF with Word2Vec

F1-score: 57.7%

Entity	Precision	Recall	f1-score	Support
ADR	0.54	0.57	0.55	1178
Drug	0.79	0.79	0.79	346
Finding	0.41	0.09	0.15	74
Disease	0.40	0.22	0.28	46
Symptom	0.17	0.03	0.04	40
Micro Avg	0.58	0.57	0.58	1684
Macro Avg	0.57	0.57	0.56	1684

For the twitter dataset we extracted 10 days of data from November 2018 which is around 100 Mb and has almost 300k tweets. Because of the size we trained and tested on truncated dataset with around 10k tweets. We have annotated this data using snomed concept file as mentioned earlier. Following are the results on LSTM-CRF based model with ELMo embeddings and CNN-LSTM-CRF with Character Embeddings:

LSTM-CRF with ELMo

F1-score: 96.0%

Entity	Precision	Recall	f1-score	Support
Symptom	0.97	0.95	0.96	2068
Micro Avg	0.97	0.95	0.96	2068
Macro Avg	0.97	0.95	0.96	2068

Analysis

The trained model performs reasonably fine although even after applying the CNN-LSTM-CRF model we still get some misclassified words and in some cases the model predicts logically correct labels even though it does not match with the actual label because of the context of the given sentence and the word being multi-label. Also some words have been categorised under multiple labels by our model due to the presence of the same in the given dataset. In some instances the model logically extend the label to include other words as well which is not present in the dataset but is logically correct. These statistics are not captured by the above given evaluation metric.

```
Word            ||True ||Pred
==============================
So              : O     O
two             : O     O
months          : O     O
ago             : O     O
the             : O     O
internist       : O     O
put             : O     O
me              : O     O
on              : O     O
Lipitor         : B-Drug B-Drug
for             : O     O
high            : B-Finding B-Disease
cholesterol     : I-Finding I-Disease
0               : O     O
```

```
Word            ||True ||Pred
==============================
This            : O     O
medication      : O     O
helped          : O     O
get             : O     O
me              : O     O
back            : O     O
to              : O     O
work            : O     O
and             : O     O
achieve         : O     O
tolerable       : O     O
pain            : B-Symptom B-Symptom
levels          : O     O
and             : O     O
mobility        : B-Symptom B-Drug
0               : O     O
```

```
Word            ||True ||Pred
==============================
58              : O     O
I               : O     O
take            : O     O
1               : O     O
/               : O     O
2               : O     O
of              : O     O
a               : O     O
20              : O     O
mg              : O     O
generic         : O     O
Simvistatin     : B-Drug B-Drug
pill            : O     I-Drug
daily           : O     O
0               : O     O
```

```
Word            ||True ||Pred
==============================
After           : O     O
I               : O     O
stopped         : O     O
taking          : O     O
Lipitor         : B-Drug B-Drug
,               : O     O
the             : O     O
pain            : B-ADR B-Symptom
gradually       : O     O
subsided        : O     O
over            : O     O
a               : O     O
9               : O     O
month           : O     O
period          : O     O
0               : O     O
```

```
Word            ||True ||Pred
==============================
recently        : O     O
pain            : B-Symptom B-ADR
has             : O     I-ADR
started         : O     I-ADR
to              : O     I-ADR
increase        : O     I-ADR
though          : O     O
,               : O     O
and             : O     O
not             : O     O
sure            : O     O
if              : O     O
I               : O     O
'               : O     O
ve              : O     O
overworked      : O     O
my              : O     O
back            : O     O
0               : O     O
```

```
Word            ||True ||Pred
==============================
Felt            : O     B-ADR
like            : O     I-ADR
a               : O     I-ADR
miscarriage     : B-Finding I-ADR
(               : O     O
I               : O     O
'               : O     O
ve              : O     O
had             : O     O
3               : O     O
of              : O     O
them            : B-Finding O
)               : O     O
0               : O     O
```

Challenges we faced

1. As the rules and features for the medical data was different from that of ordinary data, this problem was more challenging when compared to named-entity-recognition problem on normal data.
2. Twitter data is user-generated social media text, thus, it was highly disorganized and prone to inconsistencies. They contained a lot of noise apart from the required information. Filtering the noise/inconsistencies out from the tweets was a major challenge for our project as these affected the performance drastically.
3. Learning distributed representations for medical tweets. (this can overcome the weaknesses of 'bag-of-words' models)
4. We had to identify the relevant content from a given tweet. (For example, all tweets containing the keyword 'morphine' might not about the drug 'morphine')

Practical Applications

1. From the results obtained, we can get the specific details of any disease that has widely spread in a particular area.
2. Results could be analysed to find the the patient's feedback/response for a particular drug, the effectiveness of a particular drug (how far it has been successful in treatment and what are the negative points)
3. Results can be utilised by companies producing medical products for improving their sales.

Code Link to Base-line implementation

https://github.com/adisarip/medical_entity_recognition

Link to webpage and other artifacts

Data and Youtube Video link

https://drive.google.com/drive/folders/1XLysnpBP7nejFEpv1GwwqYn1u-3I83zK?usp=sharing

Web page link

https://geethika98.github.io/Medical-Entity-Recognition-on-Tweets/

References

1. BioBERT: https://arxiv.org/pdf/1901.08746v4.pdf
2. LSTM-CNN: https://arxiv.org/pdf/1511.08308.pdf
3. LSTM-CRF: https://arxiv.org/pdf/1603.01360.pdf
4. LSTM_CRF with ELMo: https://arxiv.org/pdf/1810.10566.pdf